避難エージェントのアルゴリズム解説資料

1） HIGHBROW

HIGHBROWとは，地域住民をモデル化したエージェントである．そのため最寄りの避難場所を理解していると仮定し，氾濫が発生して避難開始後，浸水しない道を通り，可能な限り最短経路で避難場所に移動するよう設定した.

HIGHBROW の実行ルールは主に 3 つの関数から成り立っている．避難に成功したエージェントを消す K_AGENT()，浸水する予定の道から安全な道に避難する I_ESCAPE ()，O_ESCAPE()，最適な避難行動をする Escape()の関数から成る．以下の図 1 に HIGHBROW における，実行ルールのフローチャートを示す.

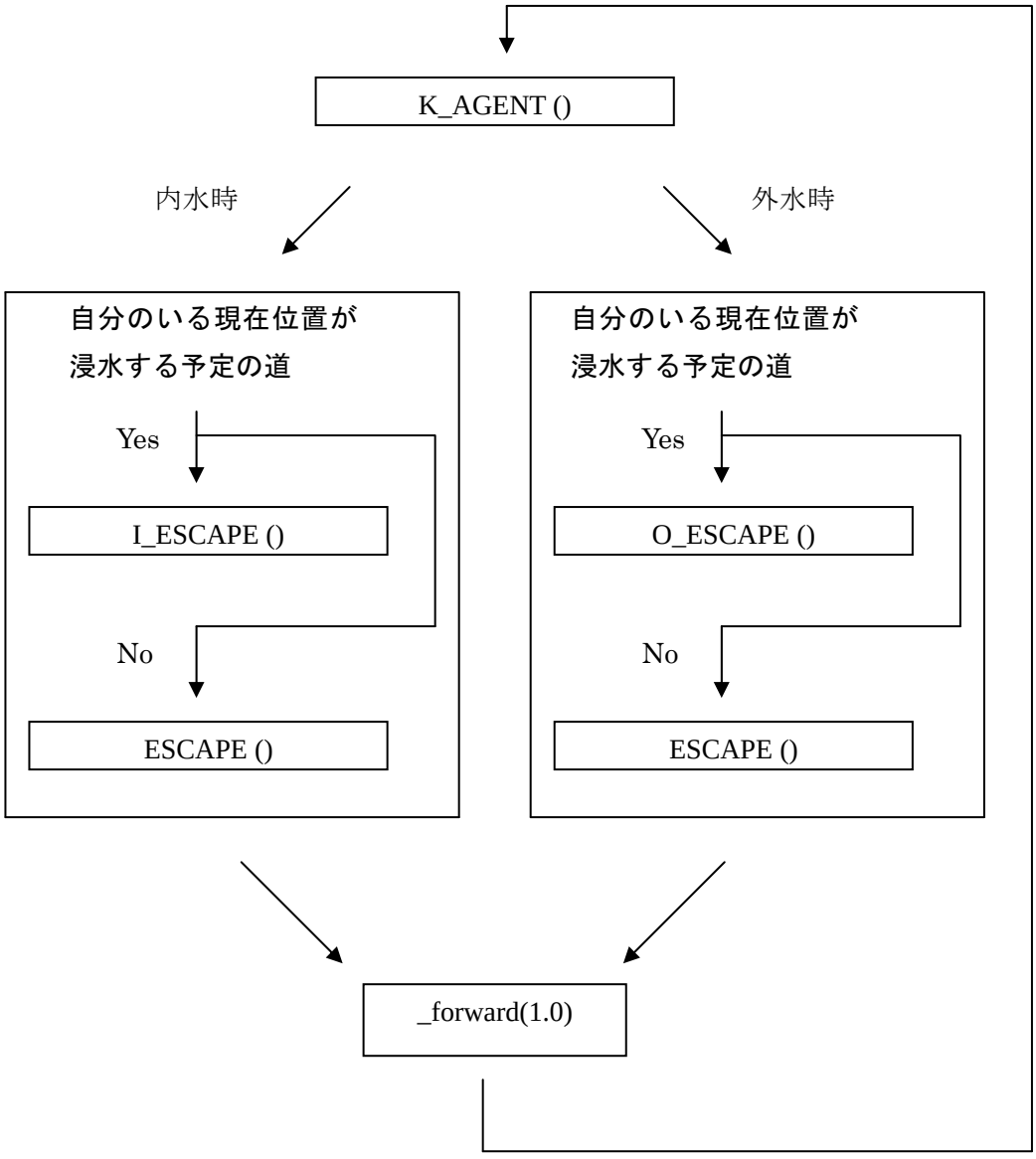

図 1 Highbrow における実行ルールのフローチャート

以下の図 2 に K_AGENT 関数における，フローチャートを示す．K_AGENT 関数は主にエージェントを削除するために構成された関数である．

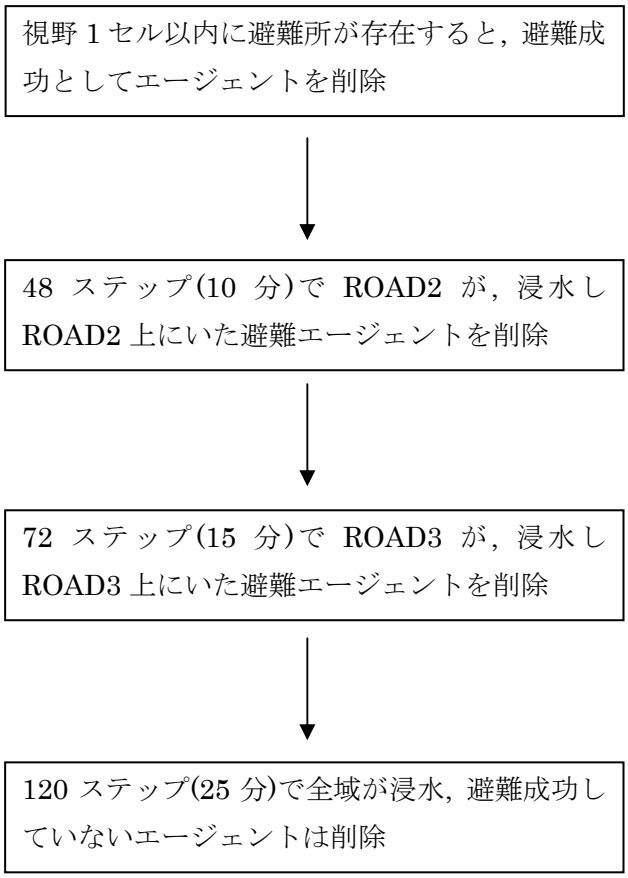

図 2 Highbrow の K_AGENT 関数におけるフローチャート

以下の図 3 に Escape 関数における，フローチャートを示す. Escape 関数は主に避難エージェントが最適に避難するための関数であり，後述する一部条件下での強制移動と普通移動に大きく分けられる．

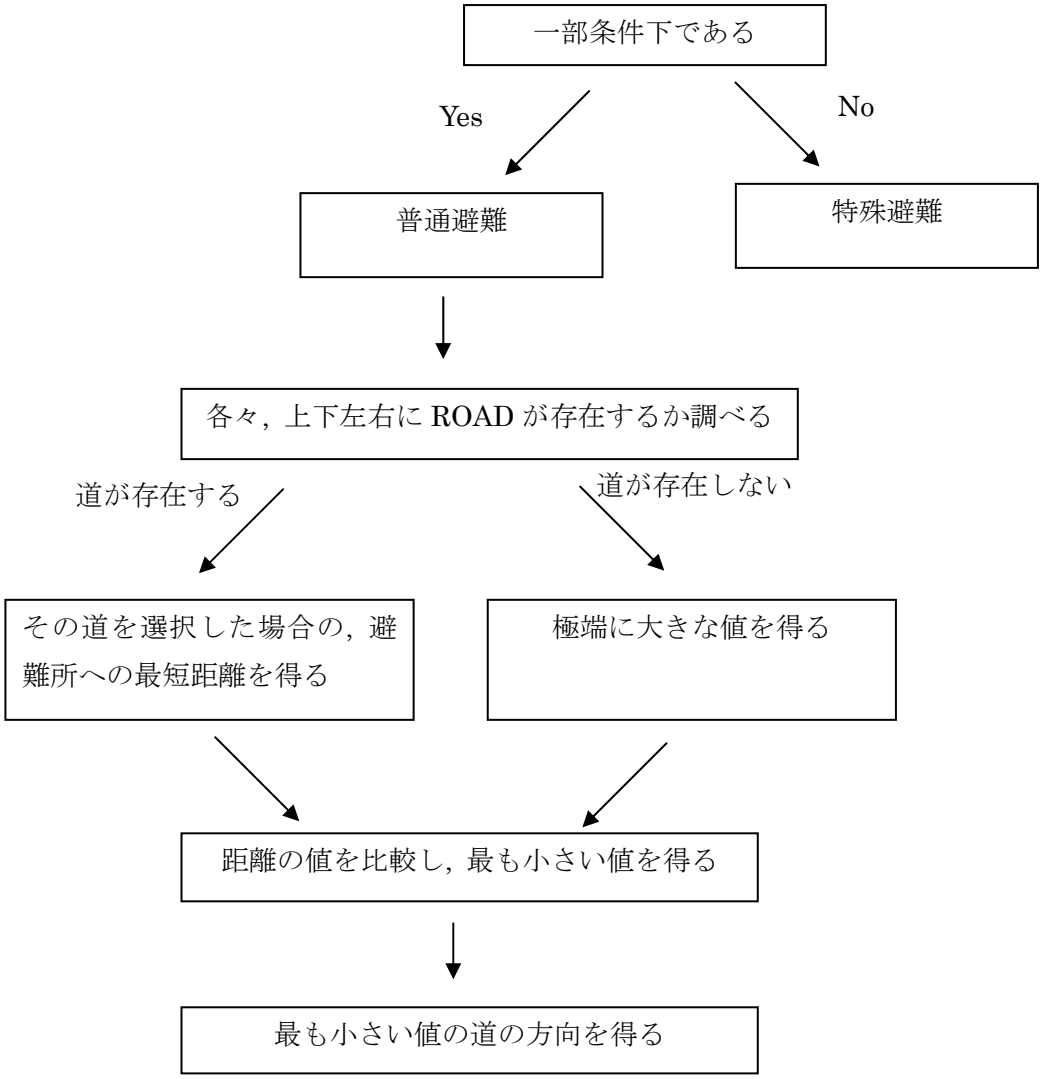

図 3 Escape 関数のフローチャート

しかしながら，一部条件下では上記のフローチャートでは進まなくなるので，補助として，強制的に進む方向を決めた．

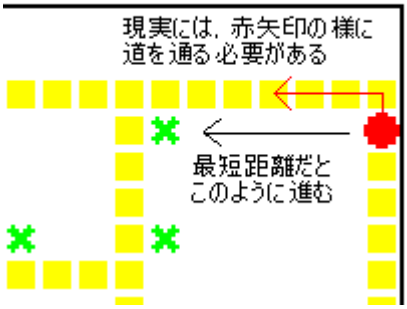

図 4 一部条件上の例

2） LOWBROW

LOWBROW は，住民エージェントのうち，洪水が外水氾濫か内水氾濫かどちらによるものか知らず，正確な道路の通行可能情報を持たないエージェントであり，避難場所までの避難経路は知らない．そのため，LOWBROW は，避難開始後ある程度ランダムで避難活動を行う．

LOWBROW の実行ルールは主に 4 つの関数から成り立っている．避難に成功したエージェントを消す K_AGENT()，避難経路を知らないのでランダムに行動する G_RND()，一時避難所から出て，避難を再開させる Out_TEMP()，Highbrow と同じアルゴリズムで避難をする I_ESCAPE ()，O_ESCAPE()，Escape()の関数．以下の図 5 に LOWBROW における，実行ルールのフローチャートを示す．LORBROW は主にエージェントの色でその行動を変化させる．

・青色　避難経路を知らない状態．LOWBROW の初期状態.
・白色　一時避難状態．一時避難所に入っているとなる．見掛け上色が白なのでエージェントは見えない.
・緑色　避難経路を知っている状態．一時避難所に入った避難エージェントは避難所までの避難経路を知っているので，最適に動く（HIGHBROW）と同じ.

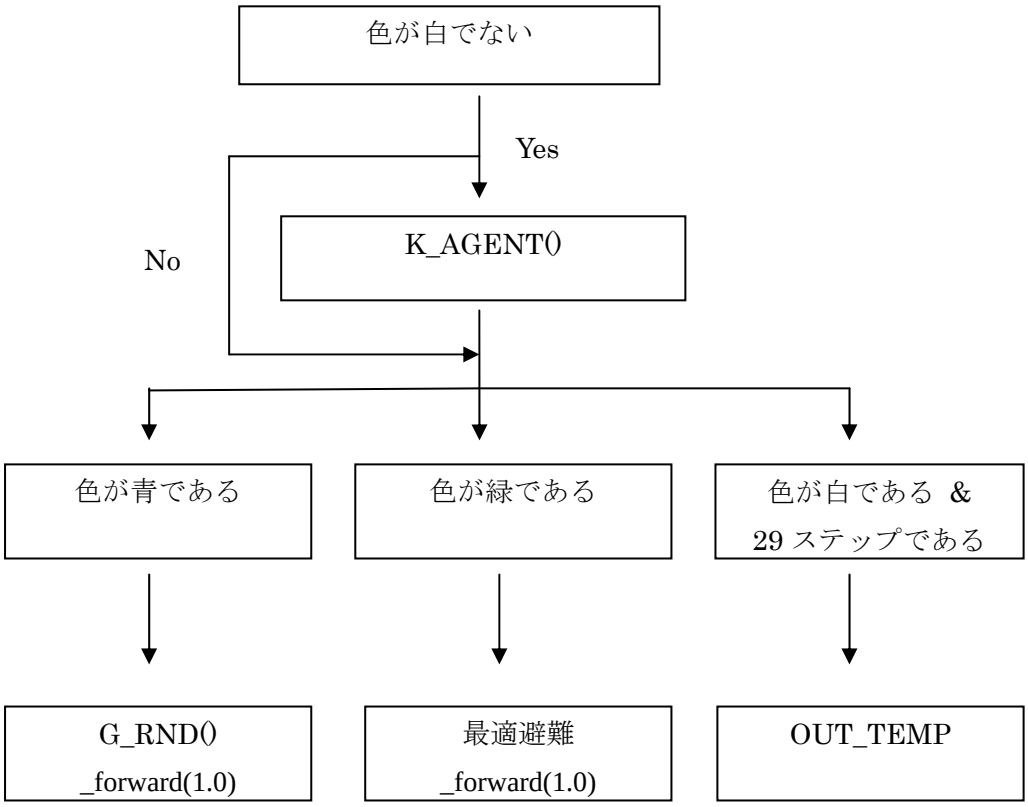

図 5 図 1 Lowbrow における実行ルールのフローチャート

以下の図 6 に LOWBROW の K_AGENT 関数における，フローチャートを示す．K_AGENT 関数は主にエージェントを削除するために構成された関数である．HIGHBROW との違いは一時避難所に避難するかである

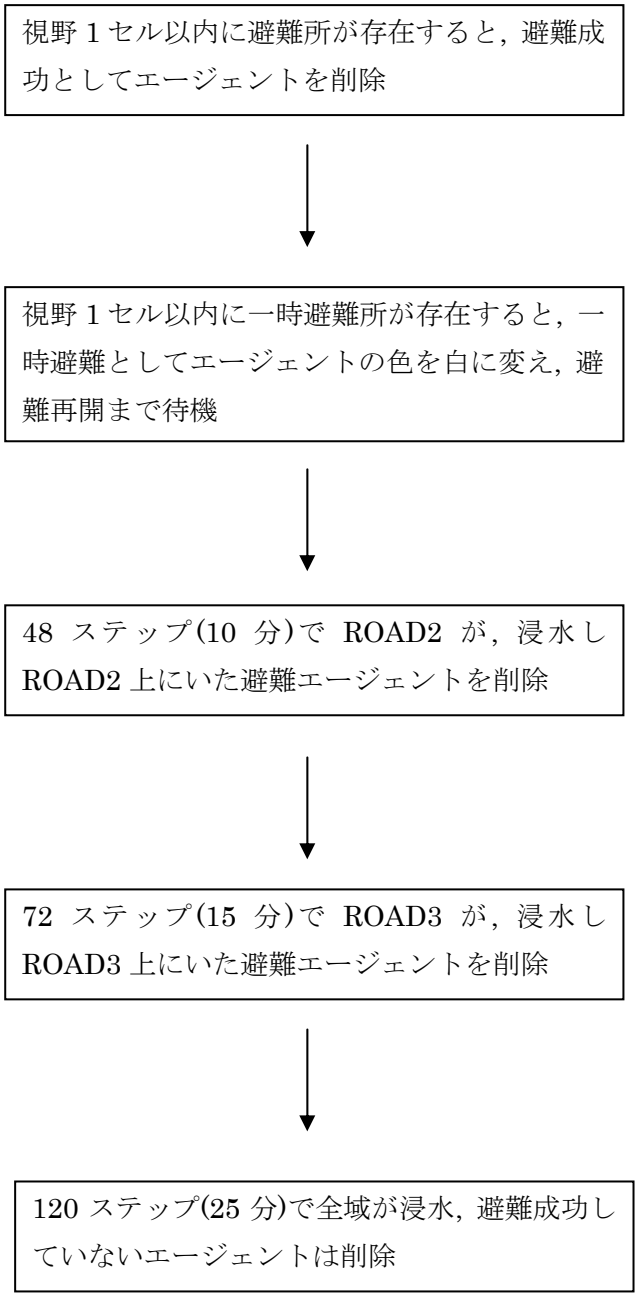

図 6 Lowbrow の K_AGENT 関数におけるフローチャート

以下の図 7 に G_RND 関数のフローチャートを示す．G_RND は避難経路を知らないエージェントの動きを制御する関数である．

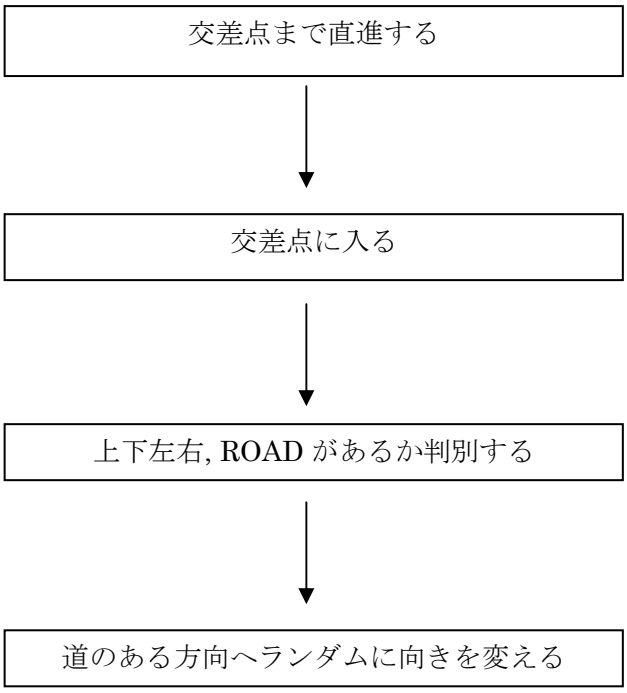

図 7 G_RND 関数におけるフローチャート

以下の図 8 に OUT_TEMP 関数のフローチャートを示す．OUT_TEMP は一時避難所に入った避難エージェントを避難再開させるための関数である．一時避難所とは元々災害が発生すると，一時的に避難し，災害情報などを整理した後．避難再開させるための場であり，今回は，一時避難所に入ると，避難経路の知らない LOWBROW でも，避難再開後は避難経路を理解できるようにした．

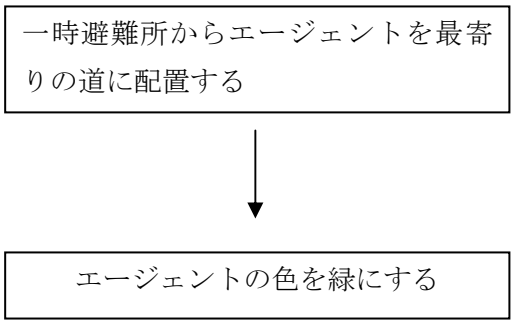

図 8 OUT_TEMP 関数におけるフローチャート

なお最適避難をする I_ESCAPE (), O_ESCAPE(), Escape()の関数のアルゴリズムは HIGHNBROW とおなじなのでここでは割愛する．